ELSEVIER

Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational and Applied Mathematics 177 (2005) 35–53

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

www.elsevier.com/locate/cam

TRIOPT: a triangulation-based partitioning algorithm for global optimization

Y. Wu, L. Ozdamar*, A. Kumar

Systems and Engineering Management Division, School of Mechanical and Production Engineering, Nanyang Technological University, 50 Nanyang Avenue, 639798 Singapore, Singapore

Received 25 October 2003; received in revised form 18 August 2004

Abstract

We propose a triangulation-based partitioning algorithm, TRIOPT, for solving low-dimensional bound-constrained black box global optimization problems. The method starts by forming a Delaunay triangulation of a given set of samples in the feasible domain, and then, it assesses the simplices (partitions) obtained for re-partitioning. Function values at the vertices of each partition are mapped into the zero one interval by a nonlinear transformation function and their aggregate entropy is calculated. Based on this entropy, partitions that hold a promise of containing the global optimum are re-partitioned according to different triangular splitting strategies, forming new partitions. These strategies are efficient in terms of the number of new function evaluations required per new partition.

A novelty in the search scheme proposed here is that once a partition narrows down to a small size, its vertices are eliminated from the available sample set. This changes global information on the best solution and triggers a re-calculation of transformed values. Hence, revised entropies change the direction of the search to new areas. The latter scheme leads to a dynamic parallel search policy which is based on an entropy cut. The tree adopts flexible breadth depending on the status of the search. In the experimental results it is demonstrated that TRIOPTs performance is compatible and often better than that of a well-known response surface methodology and two other efficient black box partitioning approaches proposed for global optimization.

Keywords: Global optimization; Triangulation; Partitioning

* Corresponding author. Tel.: +65 67905938; fax: +65 67911859.

E-mail addresses: wuyong@pmail.ntu.edu.sg (Y. Wu), mlozdamar@ntu.edu.sg, lozdamar@hotmail.com (L. Ozdamar), makumar@ntu.edu.sg (A. Kumar).

0377-0427/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2004.08.005

1. Introduction

1.1. Problem background

Black box optimization approaches are important from a practical point of view, since they deal with problems where the objective function does not have an explicit mathematical expression, and a function evaluation implies the execution or simulation of a physical experiment. In such cases, it is very costly to obtain a new function evaluation and the emphasis in performance assessment is on minimizing the number of function evaluations in the search for the global optimum. Here, we propose a black box optimization algorithm for the bound-constrained global optimization problem expressed below.

$$\text{find } \mathbf{x}^* \in \mathscr{D} \text{ such that } f(\mathbf{x}^*) \geqslant f(\mathbf{x}), \ \forall \mathbf{x} \in \mathscr{D}. \tag{1}$$

Here $\mathscr{D} \subset \mathbb{R}^n$ is the feasible domain, $\mathbf{x}$ in an n-dimensional vector, and $f : \mathscr{D} \to \mathbb{R}$ is the objective function.

Among different black box optimization methods found in the literature, two major approaches are particularly relevant to the work presented here.

The first is the broad class of partitioning approaches that have a long history of success both in continuous and discrete optimization. These methods partition $\mathscr{D}$ into sub-spaces, assess them and confine local search within the boundaries of these partitions. In order to select promising sub-spaces to re-partition and intensify the search, partitioning methods may rely on a priori knowledge about the rate of change of the function, i.e., the Lipschitz constant, (e.g., [12]), on interval bounds of the function (e.g., [11,27]), on Bayesian methods (e.g., [19]), or, on random search techniques [5] providing evidence for fuzzy assessment of partitions [7,21]. Reviews and numerical comparisons of different adaptive partitioning methods are found in [24], and [25] among many others and more recently in [22].

Most of the above-mentioned partitioning approaches, such as interval methods and Lipshitzian techniques, are not black box approaches, since they assume to have knowledge on the explicit expression of the function to be optimized or the Lipshitz constant. There are, however, black box Lipshitzian approaches that eliminate the necessity of specifying the Liptshitz constant. An example is DIRECT [16] that adopts a different perspective of Lipshitzian optimization. DIRECT views the Lipshitz constant as a weighting parameter that balances global and local search by conducting parallel partitioning on boxes that are nondominated with respect to two criteria, box size (representing unexplored areas—a global search feature) and box value (representing function fitness—a local search feature). A limitation of DIRECT is the convergence requirement of surface smoothness.

Another black box partitioning approach, Multilevel Coordinate Search (MCS) is presented in [13]. Unlike DIRECT, MCS performs nonuniform partitioning and imposes a partition bias by dividing boxes towards samples having better function values. MCS economizes on the number of new samples by collecting them from box boundaries, so that samples are shared by multiple boxes. Further, boxes are divided in one coordinate at a time. Similar to DIRECT, MCS also suffers from the limitation of smoothness requirement around the global optimum.

The second class of black box methods consist of a more recent contribution to the field of global optimization. These techniques are based on response surface methodology (RSM) and they have been initiated in [14]. RSM-based global optimization approaches are adapted from conventional methods widely used in the field of surface re-construction. In particular, radial basis functions (RBF) [26], which is also a popular technique in imaging, is selected as a preferred implementation tool in global

optimization. RBF obtains a set of samples to construct a simulated surface of the objective function through interpolation. The interpolation function consists of a linear combination of polynomial terms and special "basis function" terms that account for the nonpolynomial part of the function. The original function is estimated by a weighted set of these simpler functions where the weights are optimized for minimizing interpolation error. When RBF is adapted to global optimization, (named as RBF-O hereafter), a target value (or, estimate) for the global optimum is selected and this sets an additional condition on the interpolation problem. The next sampling point is selected so as to minimize this utility function [10].

RBF-O has been extended and commercially implemented in [4] wherein numerical tests were conducted and compare RBF-O against black box partitioning methods such as DIRECT and MCS discussed above.

Jones [15] provides an in-depth illustration of existing two- and one-stage RSM approaches in global optimization (he uses kriging [8], and RBF for surface re-construction). He classifies two-stage approaches as techniques where the next sampling location is selected based only on the optimized fit of the reconstructed surface to the given set of collected samples. The disadvantages of such approaches are illustrated on a deceptive function by using several sample selection criteria involving kriging. Since kriging enables the calculation of a confidence interval for interpolation error, these selection criteria are based on statistical lower bounds for errors, probability of improvement, or expected improvement. The author then introduces the single stage approach where the location of the targeted function value is optimized concurrently with the parameters of the RSM. Gutmann's approach, RBF-O [10] is classified under this category. The author notes that the sequential trial of multiple target values that Gutmann proposes to use in selecting the best target value is equivalent to multi-start search methods and this feature re-enforces the global component of the search and leads to convergence. Though the author strongly recommends one stage RSM approaches in global optimization, he adds that it would be computationally expensive to utilize kriging as the surface re-construction method.

1.2. Motivation of triangulation-based partitioning approach

In the above discussion on partitioning and RSM approaches, the main issue is to select the correct partition to intensify the search or to select the best location for the next sample.

Most partitioning approaches use the regular-shaped hyper-rectangle to sub-divide the domain. The "box" approach requires a significant amount of nested sub-divisions when the surface to be constructed is irregular. A widely accepted re-construction model in geographic information systems (GIS) is the triangulated irregular network (TIN). TIN is used to represent surfaces derived from irregularly spaced points. Areas of high-relief (the areas that are more important to the observer) contain a higher density of smaller triangles while areas of low-relief are represented by larger triangles. Hence, the principle of intensifying the sample density is also valid in TIN. If TIN is applied to global optimization, the main issue would be to locate the regions where a higher sample density is required to increase the probability of locating the global optimum.

Here, a triangulation-based global optimization approach, TRIOPT (TRIangulation based OPTimization), is proposed. TRIOPT does not involve the implementation of any RSM, and triangulation is only utilized because of its success in representing irregular surfaces and its property of being an economical method in sub-dividing the region with new samples. Initially, TRIOPT starts with a set of samples collected randomly from $\mathscr{D}$ or according to a pre-defined pattern. Then, $\mathscr{D}$ is partitioned by applying

Delaunay Triangulation on the sample set. Each partition (simplex) is then assessed for re-partitioning using transformed function values at its vertices. Positive assessment results in triangular re-partitioning of selected simplices. Thus, similar to other partitioning methods, TRIOPT zooms into sub-spaces where the global optimum might be located by taking as few samples as possible.

TRIOPT has an important feature that enables it to conduct parallel search of a flexible nature where the number of parallel nodes (promising sub-spaces) is dynamic. In this scheme simplices with positive assessments are re-partitioned in parallel. This flexible concurrent search is achieved by deliberately manipulating global information. The vertices of simplices whose size is lower than a given tolerance level are removed from the set of known solutions. Hence, global information on the best solution is updated every time a simplex reaches a small size and some of its vertices that are not shared by neighboring simplices are removed from the collection of samples. This affects the status of pending simplices in the search tree, because the partition assessment scheme does not use the original function values of vertices, but their transformed values mapped into the zero one interval. This transformation uses global information on best and worst solutions.

With its efficient triangular partitioning schemes that economize the number of new samples taken, and this novel and flexible parallel search strategy, TRIOPTs performance is compatible with existing black box methods discussed above, and often, better.

In the following sections, TRIOPT is discussed in detail with all its features and, in the numerical results, a performance comparison of TRIOPT, DIRECT, MCS and RBF-O is provided.

2. Algorithm description

2.1. Triangulation-based partitioning

If we assume that a partition is represented by samples located at its vertices, then, we can define partitioning efficiency, er, as the ratio of the number of new partitions generated by splitting a parent partition to the number of new vertices required:

$$\text{er} = \frac{\text{No. of new partitions}}{\text{No. of new vertices required}}. \tag{2}$$

Accordingly, two new vertices are required when a box in 2D is bisected while one vertex is required for bisecting a triangle (simplex) from one of its edges or for trisecting it from an interior point. In general, er is $(n+1)/1, 2/1, 2/2^{n-1}$ for interior simplex, edge, and box bisection, respectively. This is shown in Fig. 1(a)–(c), in 2D. Obviously, interior simplex partitioning is the most efficient one.

TRIOPT exploits the efficiency of simplicial partitioning by constructing a triangulation of domain $\mathscr{D}$ on a given set of initial samples. Many triangulation algorithms are available in the literature (e.g., [1,29]). We now proceed to a formal definition of triangulation.

Definition 1. A *simplex* $\mathscr{S}$ in $n\,D$ is the convex hull of any $n+1$ noncoplanar (noncolinear in 2D) points in n dimensions.

Definition 2. A *facet* $\mathscr{F}$ in $n\,D$ is the convex hull of any n vertices of a simplex in n dimensions.

Definition 3. An *edge* $\mathscr{E}$ in $n\,D$ is a line connecting two vertices of a simplex in n dimensions.

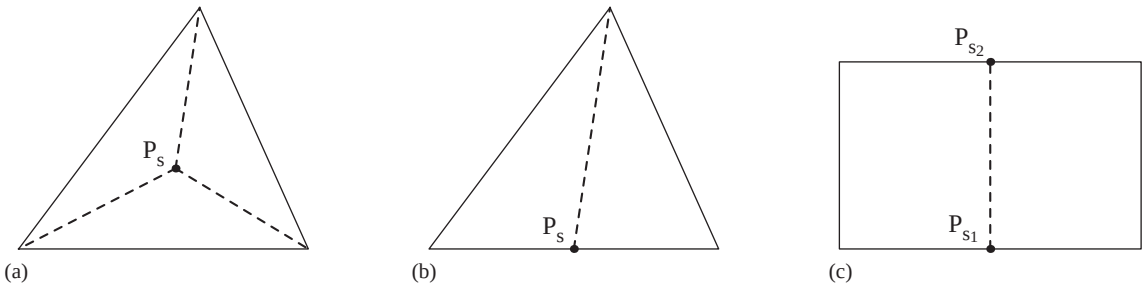

Fig. 1. An illustration of the efficiency of simplex partitioning.

Definition 4. A set $\triangle = \{\mathscr{S}_1, \ldots, \mathscr{S}_M\}$ of simplices $\mathscr{S}_j$, $j = 1, \ldots, M$, is called a *triangulation* of a point set $\mathscr{V}$ provided that:

- The points in $\mathscr{V}$ are the vertices in $\triangle$;
- Any pair of simplices $\mathscr{S}_i$ and $\mathscr{S}_j$ intersect at most at one common vertex, or, along a common edge or a common facet;
- The union of the simplices is a connected set in $\mathbb{R}^n$, which means there is always a path of facets or a path of edges connecting any two vertices in the triangulation;
- The union of the simplices is equal to Ω, the convex hull of point set $\mathscr{V}$.

A triangulation can be viewed as a *partition* of a given domain based on a sample set $\mathscr{V}$. In general, a partition is defined as follows.

Definition 5. Let M be a closed set in $\mathbb{R}^n$ and let I be a finite set of indices. The union of closed subsets M, $M_i : i \in I$, is said to be a *partition* of M if

$$M = \bigcup_{i \in I} M_i, \quad \neg M_i \cap \neg M_j = \emptyset, \quad i, j \in I, \quad i \neq j,$$

where $\neg M_i$ excludes the boundary of M_i.

Among various kinds of triangulations, Delaunay triangulation is the most widely used and researched. Its definition is given below.

Definition 6. Let $\triangle$ be a triangulation of the convex hull of $\mathscr{V}$. Then $\triangle$ is a *Delaunay triangulation* if and only if for each simplex $\mathscr{S}_j \in \triangle$, the interior of the (hyper-)circumcircle of $\mathscr{S}_j$ contains none of the vertices in $\mathscr{V}$.

Fig. 2 shows a Delaunay and a nonDelaunay triangulation on a set of same four points in 2D.

Considerable effort has gone into the design of algorithms for the construction of a Delaunay triangulation, because the latter is angle optimal (it avoids small angles) and leads to more regular simplices. The Quickhull algorithm [3] and DeWall algorithm [6] are such examples. Those two algorithms have the ability to form Delaunay triangulation in $\mathbb{R}^n$ space. Here, the Quickhull algorithm is adopted to generate

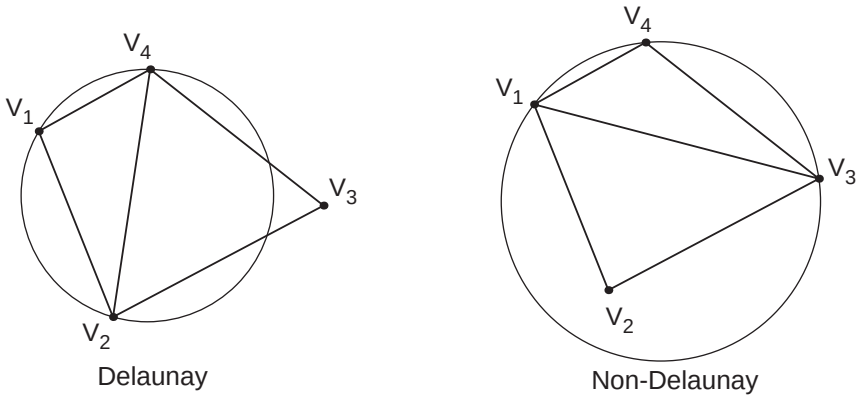

Fig. 2. Delaunay vs. nonDelaunay triangulation.

the initial partition of the domain. Once the initial partition is obtained, each simplex is subjected to an assessment which leads to the decision for re-partitioning.

2.2. Simplex assessment

TRIOPT is based on a 're-partition and intensify' strategy. Promising simplices of the triangulated domain $\mathscr{D}$ are re-partitioned based on their *entropies*.

First, the function value $f(\mathbf{x})$ of each sample $\mathbf{x} \in \mathscr{V}$ forming the triangulation is calculated and sorted in descending order. Next, $f(\mathbf{x})$ are transformed by a mapping function into the [0.1, 1.0] interval. This transformation is carried out by defining a threshold t selected from the range of function values in $\mathscr{V}$ defined as $[f_{\min}, f_{\max}]$:

$$t = f(\mathbf{x}_{[i]}), \tag{3}$$

where $\mathbf{x}_{[i]} \in \mathscr{V}$ is the ith (the index may be selected by the user) sample in the ordered set $\mathscr{V}$.

In this scheme, t represents a 'yardstick' where $f(\mathbf{x}) \geqslant t$ are treated as peaks and $f(\mathbf{x}) < t$ as valleys. The transformation function $\ddot{f}$ utilized here is expressed in the following equation:

$$\ddot{f} = \ddot{f}(\mathbf{x}) = \ddot{f}(f(\mathbf{x}), t) = \begin{cases} 0.1 + 0.4\left(1.0 - \dfrac{t - f(\mathbf{x})}{t - f_{\min}}\right)^{p} & \text{if } f(\mathbf{x}) \leqslant t, \\ 0.5 + 0.5\left(\dfrac{f(\mathbf{x}) - t}{f_{\max} - t}\right)^{q} & \text{if } f(\mathbf{x}) > t. \end{cases} \tag{4}$$

Here, p and q are positive deflation and inflation parameters, $p > 1.0$ and $q < 1.0$. This function deflates original function values below the threshold t and inflates the ones above, leading to a more clear classification in partition assessment. $\ddot{f}$ is a two-piece nonlinear function with a slope cut-off at threshold, t. The lower bound on $\ddot{f}$ is related to simplex splitting procedures and will be explained later.

After the calculation of $\ddot{f}$ for all samples, an entropy E_j is computed for each simplex S_j by considering only its vertices. The following equation is used to calculate E_j:

$$E_j = \begin{cases} \dfrac{1}{k} \displaystyle\sum_{\mathbf{x}\in\mathscr{S}_j} \ddot{f}(\mathbf{x}) \exp(1.0 - \ddot{f}(\mathbf{x})) & \text{if } \exists \ddot{f}(\mathbf{x}) \geqslant \alpha, \\ 0.0 & \text{otherwise,} \end{cases} \tag{5}$$

where, each vertex $\mathbf{x}$ contributes to the entropy if $\ddot{f}(\mathbf{x}) \geqslant \alpha$. k is the number of vertices in a simplex satisfying this condition. The cut α on $\ddot{f}$ is a user-defined parameter in the unit interval. It prevents inferior vertices in a simplex from affecting the potential reflected by better ones. E_j indicates the degree of attraction that a simplex holds for re-partitioning. The expression given here is of a multiplicative form and it is adapted from [23]. *All* simplices with $E_j \geqslant \beta$, (an entropy cut in the unit interval), are selected for re-partitioning in parallel (the minimum value for β is $\alpha \exp(1 - \alpha)$). Thus, depending on the status of global information ($f_{\max}$, $f_{\min}$, t) updated during partitioning iterations, the number of parallel nodes scanned differs. Next, we explain how global information is manipulated to enable such a flexible tree.

2.3. Global search instrument in TRIOPT

In solving any global optimization problem, considerable effort should be allocated to global search while exploiting local search around "good" solutions. That is, search localizing into promising sub-regions should not be over-indulgent to ensure that other areas are not overlooked. TRIOPTs parallel re-partitioning scheme contributes to global search, however, a further mechanism exists to emphasize this global feature. It works as follows: once the size of a simplex, $m(\mathscr{S}_j)$, reaches a tolerance level, δ, it is removed from the set of pending simplices and all its vertices not shared by its neighbors, $\{\mathbf{x} \in \mathscr{S}_j \wedge \mathbf{x} \notin \mathscr{S}_{i\neq j}\}$, are discarded from the set $\mathscr{V}$. Hence, $f_{\max}$, $f_{\min}$ and threshold value t are possibly modified to result in readjusted transformed values $\ddot{f}$. In this sense, the threshold t is dynamic and changes adaptively according to the status of the search. Hence, the same location $\mathbf{x} \in \mathscr{D}$ can be classified as a peak or as a valley during different phases of the search. This enables a flexible search tree with varying numbers of parallel nodes, where less attractive nodes (subspaces) are also explored in the natural course of partitioning depending on the level of t.

2.4. Re-partitioning simplices

Two partitioning schemes are defined: *interior-simplex* and *on-edge*. In both cases, the specific location of the new vertex is calculated as the weighted average of the simplex's vertex locations, with the weights being their transformed function values, $\ddot{f}$.

Interior-simplex splitting: Interior-simplex splitting is carried out by selecting a point $\mathbf{z}_j$ inside the simplex $\mathscr{S}_j$ and splitting the parent simplex into $n + 1$ new simplices. Each child simplex is formed by connecting the newly generated point $\mathbf{z}_j$ to each of the $n + 1$ facets. The coordinates of $\mathbf{z}_j$ are calculated as follows:

$$\mathbf{z}_j = \sum_{\mathbf{x}\in\mathscr{S}_j} \left[\mathbf{x} \cdot \frac{\ddot{f}(\mathbf{x})^r}{\sum_{\mathbf{x}\in\mathscr{S}_j} \ddot{f}(\mathbf{x})^r} \right]. \tag{6}$$

Here, r is a nonnegative coefficient affecting the bias towards each vertex.

On-edge splitting: On-edge splitting is carried out by selecting a point $\mathbf{z}_j$ on one of the edges that connects two vertices of the simplex. (Here, the longest edge is selected for the split with the aim of forming more regular simplices, however, other criteria could be used.) This operation splits the original simplex into two new simplices. The point $\mathbf{z}_j$ is selected in a similar manner by taking the convex combination of vertex coordinates forming the edge.

In case a vertex has a transformed value of 0.0, it cannot contribute to the location of $\mathbf{z}_j$, and, interior-simplex splitting selects a point on one of the facets. On the other hand, on-edge splitting fails. By setting the minimum transformed value to 0.1, such degenerations and splitting failures are avoided. This explains the lower bound on $\ddot{f}$ in Eq. (4).

Obviously, interior-simplex is more efficient than on-edge partitioning. However, a persistent application of interior-simplex partitioning can result in very irregular simplices. On-edge splitting is applied as a compensation. The decision on how to partition a simplex is made upon the following definition of simplex regularity.

Definition 7. Let L_{ju} denote the shortest edge length of a simplex $\mathscr{S}_j$. If the length L_{ji} of each edge satisfies $L_{ju} \leqslant L_{ji} \leqslant 2L_{ju}$, then $\mathscr{S}_j$ is called a *regular simplex*.

The following rule controls re-partitioning in TRIOPT:

Simplex partitioning rule: *If a simplex $\mathscr{S}_j$ is regular, re-partitioning is carried out by interior-simplex splitting, otherwise on-edge splitting is applied.*

2.5. The algorithm

The following notation is required for describing the algorithm:

$\mathscr{W}$: List of all pending simplices that require a decision for re-partitioning;
$\mathscr{P}$: Final list of simplices to be reported as having potential to enclose $\mathbf{x}^*$;
$\mathscr{S}_c$: Current simplex being processed;
$m(\mathscr{S}_c)$: size of current simplex;
$\mathscr{N}$: List of child simplices obtained by splitting $\mathscr{S}_c$;
$\mathscr{T}$: List of simplices stored temporarily;
$\succ$: Insert symbol, $B \succ A$ means insert element B into list A;
$\prec$: Remove symbol, $B \prec A$ means remove element B from list A;
$H(A)$: Get the first element of list A;
$N(A)$: Get the next element of list A.

In Fig. 3 the flowchart of TRIOPT is presented. In the "Initialization" step, a set of samples are generated and their function values evaluated. Each sample is inserted into the point list $\mathscr{V}$ and a threshold t is selected resulting in transformed function values $\ddot{f}(\mathbf{x})$ (See Fig. 4).

After the initialization, a triangulation is formed on $\mathscr{V}$ and a working simplex list $\mathscr{W}$ is created by linking all the simplices generated.

Then, the entropies of simplices in $\mathscr{W}$ are calculated. A simplex $\mathscr{S}_c$ is selected from $\mathscr{W}$. If its size is lower than δ, then it is added to the set $\mathscr{P}$ of simplices to be reported as potential, and vertices not shared with other simplices are removed from set $\mathscr{V}$. Otherwise, if $E_{\mathscr{S}_c} \geqslant \beta$, then the simplex $\mathscr{S}_c$ is removed from $\mathscr{W}$, partitioned, and its child simplices held in a temporary list, $\mathscr{T}$. The new vertex $\mathbf{z}_{\mathscr{S}_c}$ is added

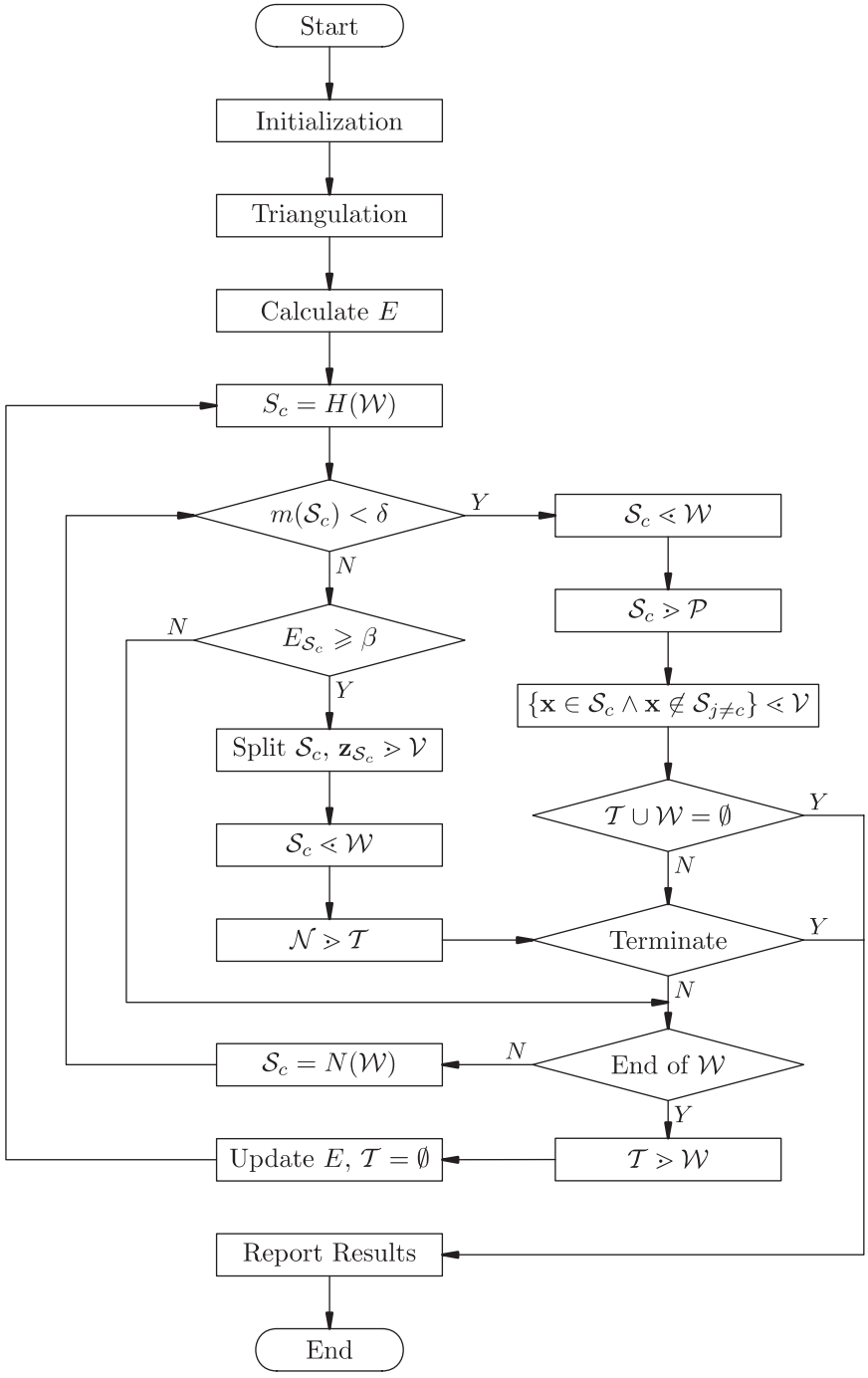

Fig. 3. Flowchart of TRIOPT.

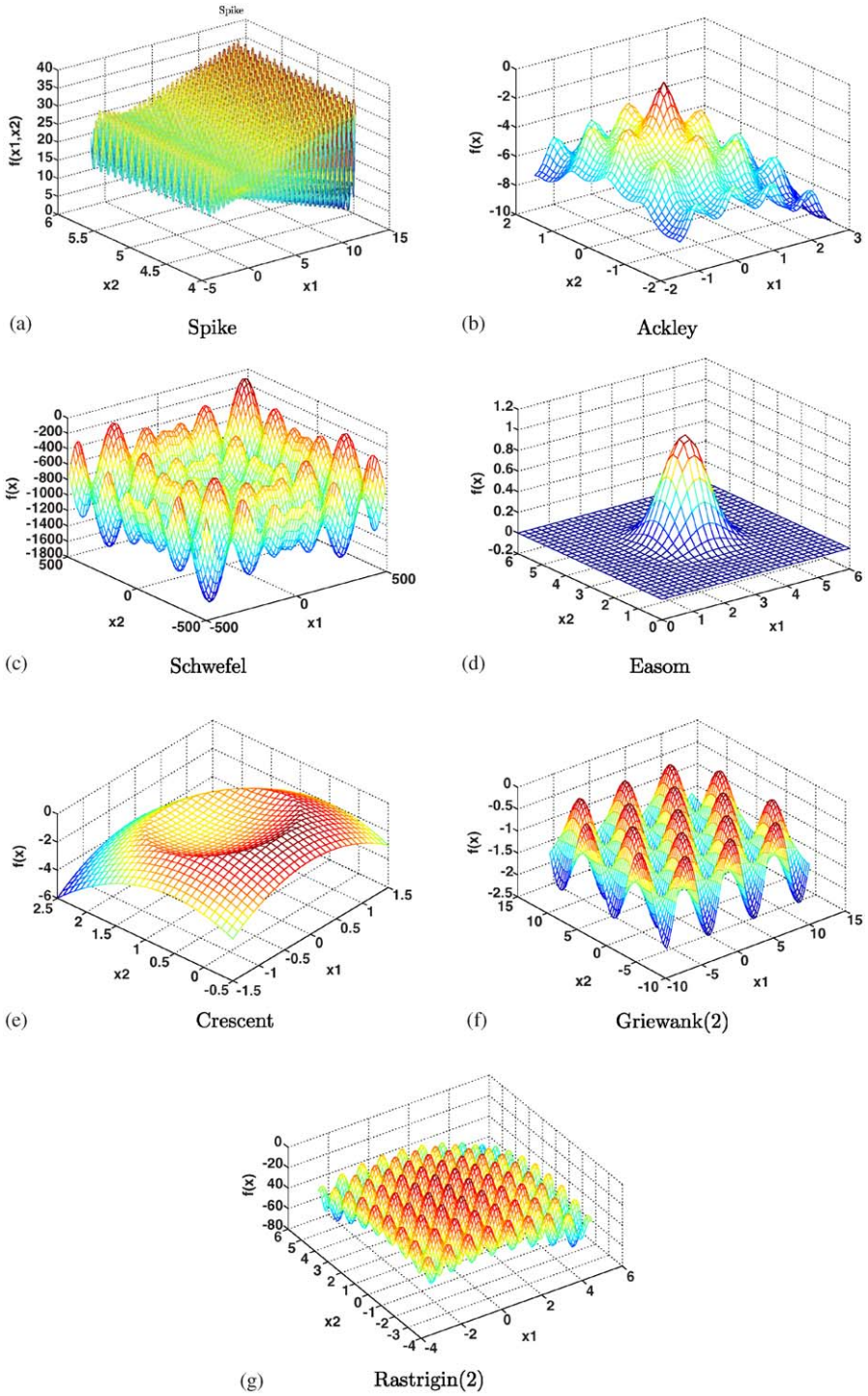

Fig. 4. Surface plots of test functions. (a) Spike; (b) Ackley; (c) Schwefel; (d) Easom; (e) Crescent; (f) Griewank (2); (g) Rastrigin(2).

to set $\mathscr{V}$. If the simplex $\mathscr{S}_c$ is not split, then it remains in $\mathscr{W}$. This step is repeated until all $\mathscr{S}_j \in \mathscr{W}$ have been evaluated leading to one of these three actions: store in $\mathscr{P}$, split or preserve. After completing the scan of all simplices in $\mathscr{W}$, any child simplices stored temporarily in $\mathscr{T}$ are added to $\mathscr{W}$. A new scanning cycle starts by flushing $\mathscr{T}$ and updating simplex entropies. New function evaluations obtained by re-partitioning and vertices discarded in the last cycle might have changed $f_{\max}$, $f_{\min}$, t, and, hence $\ddot{f}$.

TRIOPT can continue partitioning until $\mathscr{W}$ becomes empty implying that the domain $\mathscr{D}$ is partitioned into simplices with $m(\mathscr{S}_j) < \delta$. Since this is not possible in practice, a termination criterion has to be embedded. This criterion might be the maximum number of function evaluations (number of physical experiments conducted) or the percentage improvement obtained in the best function value at the end of each scanning cycle.

If, in any scanning cycle, none of the pending simplices satisfy the entropy cut condition, TRIOPT can stop prematurely. In this case, the simplex with the maximum $E_{\mathscr{S}_j}$ is partitioned no matter how small its entropy is. (This exceptional situation is not illustrated in the flowchart.) The latter policy helps the algorithm to carry on since new vertices are introduced and eventually the topology of available simplices and their entropies change.

2.6. Convergence of TRIOPT

Suppose the set of near-optimal solutions, ω, is composed of a small hull of solutions enclosing $\mathbf{x}^*$. Let the size of the basin of attraction be denoted as $m(\omega)$.

An algorithm is defined to be *convergent* if it has the capability of locating any $\mathbf{x} \in \omega$.

Theorem. *TRIOPT is a convergent algorithm given that it is not terminated by any criterion other than the simplex size tolerance,* $\delta \leqslant m(\omega)$.

Properties that make TRIOPT a convergent algorithm are as follows:

(i) Since the simplex size tolerance $\delta > 0$, the number of iterations required to move away from a local basin of attraction is finite;

(ii) Suppose t and t' are threshold values calculated just before and just after a cycle where a simplex reaches the size tolerance δ and its unshared vertices are removed. Then, $t' \leqslant t$. Let κ be the number of removed simplices holding local optima, and let τ be the number of such local optima in $\mathscr{V}$. As $\kappa \to \tau$, then, in a finite number of iterations, $t' \to f_{\min}$ because t' is reduced in discrete steps. In the worst case, even if the simplex containing $\mathbf{x}^*$ does not contain a vertex in ω, at this point, it becomes eligible for nested partitioning, and that eventually leads to a convergent sequence of samples.

3. Numerical studies

3.1. Comparison with other black box approaches and implementation details

Three efficient black box optimization algorithms are selected to form a comparison basis for TRIOPT. Their implementation details are discussed below.

DIRECT [16]: The module *glbSolve* of the commercial software Tomlab^{TM} (*http://www.tomlab.biz*) is utilized here. As discussed previously, DIRECT is a "black-box" optimization algorithm that partitions the search space and ranks boxes according to a criterion that measures the rate of change in function

value versus rate of change in box width. Multiple boxes are trisected simultaneously (each with two new function evaluations) if they are nondominated with respect to these two criteria. Since DIRECT does not involve any specific parameters, it is applied here with the default settings in *glbSolve*.

RBF-O [4,10]: This algorithm is implemented by Tomlab^{TM} *rbfSolve* module. As a brief reminder, RBF-O is an adaptation of RBF. It selects the next point to sample by specifying a target estimate of the global optimum in each cycle. This target function value is changed a number of times and the best radial basis minimizer is selected as the next sampling location. Here, the settings of RBF-O in *rbfSolve* are given as follows. The initial set of samples are taken by Latin Hypercube design (the corner points design results in an inferior performance). The number of different target values taken in one cycle is set to the default value of 4 and the setting "replacement by median" is implemented. Since the thin plate and cubic spline results are very similar, only the cubic spline results are summarized here.

MCS [13]: MCS takes samples at the boundary of boxes according to a golden section type of splitting where one new point may be shared by two or three boxes. Partitioning can be carried out by selecting the coordinate with the least number of historical splits or by the coordinate that produces the maximum expected gain. The partitioning point is determined by taking two points and the base point to form a quadratic interpolation whose minimizer becomes the partitioning point. MCS can be accompanied by a local search after a box is split $s_{\max}$ times. Here, MCS is implemented by using the Matlab code provided in *http://www.mat.univie.ae.at/~neum/software/mcs/*. The local search option is not used, because none of the partitioning strategies compared here includes local search. The most important parameter in MCS is $s_{\max}$, the depth of the tree, and this parameter is set to three values, $5n + 10$ (recommended one), $25n + 10$ and $50n + 10$, respectively, and these are labelled as MCS1, MCS2 and MCS3 in the summary of results. It is necessary to increase this parameter when MCS cannot converge to the global optimum.

These three methods form a good basis of comparison for TRIOPT. DIRECT and MCS are well known for their partitioning efficiency and RBF-O is an algorithm with a different perspective.

Implementation details: The details of implementation for TRIOPT are as follows: TRIOPT starts with a given set of samples collected from $\mathscr{D}$. In this implementation, a sample is taken at each vertex of the hypercube $\mathscr{D}$ and one additional sample is taken at its mid-point. (The minimum number of samples for a full coverage of $\mathscr{D}$ by triangulation is equal to the number of vertices defining $\mathscr{D}$). The simplex size tolerance level, δ, is set to $(\prod_{i=1}^{n} l_i)/(500n)$ where l_i is bound length of the ith coordinate. We conducted a parametric analysis of α, which shows that, any value of α greater than 0.6 does not change the results. The parameter β is taken at its minimum value, $\alpha \exp(1-\alpha)$. We omit the table for this analysis due to lack of space. Hence, the results presented here are obtained with $\alpha = 0.7$. The threshold value, t, is set by the third sample in the ordered sample list (i.e., the third sample below $f_{\max}$). It is preferred to have a threshold value closer to $f_{\max}$ to control the number of parallel re-partitioning. In any case, $f_{\max}$ is usually pulled down whenever some samples are removed from the list. The threshold also reduces accordingly.

Performance assessment: The criterion for performance assessment is the number of function evaluations that a method takes to converge to a near optimal solution that is within a small percentage below the global optimum. The formula is given below:

$$\begin{aligned} &\frac{|f^* - f|}{|f^*|} < \varepsilon \quad \text{if } f^* \neq 0.0, \\ &|f^* - f| < \varepsilon \quad \text{if } f^* = 0.0. \end{aligned} \tag{7}$$

The relative error ε is set to 1.0% and 0.01% in these sets of experiments.

Table 1
Summary of test functions

Name	Dim.	Imax.	Reference	Complexity
Spike	2	38.85	[18]	M2
Ackley	2	0.0	[2]	M1
Schwefel	2	0.0	[28]	D1
Easom	2	1.0	[9]	D1
Crescent	2	0.0	[17]	M2
Griewank(2)	2	0.0	[31]	M2
Griewank(3)	3	0.0	[31]	M2
Griewank(4)	4	0.0	[31]	M2
Rastrigin(2)	2	0.0	[31]	M2
Rastrigin(3)	3	0.0	[31]	M2
Rastrigin(4)	4	0.0	[31]	M2
Rastrigin(5)	5	0.0	[31]	M2

3.2. The test functions

The test functions utilized here are well-known highly multi-modal functions, two of which are haystack types (Spike and Rastrigin), one nondifferentiable but smooth (Crescent), some symmetric with no trend towards the global optimum (Easom, Griewank, Ackley), and one deceptive (Schwefel). The boundaries of some symmetric test functions where the global optimum lies at the origin are shifted such that the optimal solution is not found trivially.

The mathematical expressions of the test functions and their brief properties are listed in the Appendix and their figures provided. They are also summarized in Table 1 along with their references. In [20], these test functions (except the Crescent) are classified according to the complexity categories proposed in [30]. As indicated in Table 1, the complexity levels of these problems range from moderate (M1) to difficult (D1) based on properties such as the number of local optima, and embedded or isolated global optimum.

3.3. Numerical results

Table 2 shows the number of function evaluations needed to solve all test problems within a 1.0% and a 0.01% tolerance below the global optimum. In Table 2, "—" indicates the corresponding method cannot solve the problem within the given accuracy level. For RBF-O this implies that the interpolation is stuck at a local optimum and cannot generate new samples, and for MCS, it implies that the maximum level of the tree is insufficient to achieve the desired accuracy.

Table 3 provides the CPU times related to all experiments. The CPU speed of the PC utilized is P4 2.0 GHz with 256M RAM. We observe that RBF-O spends considerable time with matrix inversion, and that DIRECT is comparatively faster as compared to MCS. TRIOPT is significantly faster than other methods, due to the fact that triangulation is carried out only once and entropies are re-calculated once per complete scanning cycle.

For 1.0% accuracy (Table 2), it is observed that TRIOPT performs better than the other methods in Easom, Griewank (2) and all Rastigin variants. Considering Spike and Ackley, MCS1 performs best, and in Schwefel and Griewank (4), DIRECT solves the problem with the least function evaluations.

Table 2
Function evaluations required for 1.0% and 0.01% accuracy of test functions

Function	TRIOPT	RBF-O	DIRECT	MCS1	MCS2	MCS3
1.0% *accuracy*						
Spike	111	—	10803	82	412	412
Ackley	21	32	93	18	58	112
Schwefel	1907	568	191	—	1820	3872
Easom	21	26	39	33	91	174
Crescent	6	100	5	—	261	510
Griewank(2)	22	37	81	42	140	277
Griewank(3)	133	44	145	—	684	1481
Griewank(4)	866	—	169	—	2214	4624
Rastrigin(2)	107	443	235	—	4055	8770
Rastrigin(3)	267	—	1627	—	—	—
Rastrigin(4)	206	—	7479	—	—	—
Rastrigin(5)	1792	—	29467	—	—	—
0.01% *accuracy*						
Spike	215	—	12423	—	1938	3951
Ackley	67	40	271	18	58	112
Schwefel	1960	571	319	—	1820	3872
Easom	21	27	95	—	92	175
Crescent	6	377	5	—	3142	6360
Griewank(2)	51	—	711	—	144	281
Griewank(3)	133	54	9969	—	690	1485
Griewank(4)	9989	—	35059	—	2220	4630
Rastrigin(2)	386	450	323	—	4254	9225
Rastrigin(3)	311	—	1947	—	—	—
Rastrigin(4)	206	—	9563	—	—	—
Rastrigin(5)	1794	—	35077	—	—	—

RBF-O cannot solve haystack type of functions (Spike and Rastrigin), it also fails to determine quickly the edge of the circular crater on which the global maximum of Crescent (nondifferentiable function) lies. RBF-O is very sensitive to surface smoothness, because interpolation tends to smooth out the extremities, and the new sample is selected according to the interpolated surface. Therefore, when the surface is too rough RBF-O shows a strong tendency to oscillate. The method is also sensitive to dimensionality.

DIRECT also suffers from nonsmooth surfaces as it has to partition the domain into quite small subregions to cover very narrow basins of attraction. It might also lose its discrimination power among boxes when the function is symmetric.

It is observed that in Table 2 the performance of MCS depends on the maximum level of the tree, $s_{\max}$ and that sufficient depth should be allowed to guarantee convergence. However, increasing this parameter leads to excessive computations. In problems that it can converge, using the recommended default $s_{\max}$, MCS1 identifies the global optimum quite quickly (Spike and Ackley), but it fails to converge to the optimum in Schwefel and Crescent, where smaller boxes need to be generated to capture the required accuracy. In problems with higher dimensions, $s_{\max}$ needs to be increased and this in turn raises the number of function evaluations. It is observed that when MCS converges, the number of computations

Table 3
CPU time (s) for 1.0% and 0.01% accuracy

Function	TRIOPT	RBF-O	DIRECT	MCS1	MCS2	MCS3
1.0% *accuracy*						
Spike	0.010	—	20.63	0.766	1.750	3.733
Ackley	0.006	5.984	0.187	0.375	0.594	1.094
Schwefel	0.180	962.3	0.203	—	15.20	42.49
Easom	0.007	2.969	0.172	0.453	0.812	1.532
Crescent	0.006	50.94	0.210	—	1.969	4.547
Griewank(2)	0.006	8.313	0.188	0.514	1.157	2.391
Griewank(3)	0.014	7.703	0.234	—	6.109	17.69
Griewank(4)	0.196	—	1.094	—	22.27	70.36
Rastrigin(2)	0.011	591.0	0.250	—	35.38	95.63
Rastrigin(3)	0.040	—	0.797	—	—	—
Rastrigin(4)	0.300	—	3.140	—	—	—
Rastrigin(5)	25.23	—	16.09	—	—	—
0.01% *accuracy*						
Spike	0.020	—	25.95	—	13.91	42.83
Ackley	0.008	9.860	0.250	0.375	0.594	1.094
Schwefel	0.190	989.5	0.328	—	15.20	42.55
Easom	0.007	3.281	0.187	—	0.815	1.532
Crescent	0.006	302.4	0.223	—	27.63	75.41
Griewank(2)	0.007	—	0.547	—	1.109	2.391
Griewank(3)	0.014	15.05	11.16	—	5.828	17.78
Griewank(4)	13.10	—	127.6	—	22.34	70.84
Rastrigin(2)	0.030	617.7	0.281	—	36.01	101.8
Rastrigin(3)	0.047	—	0.969	—	—	—
Rastrigin(4)	0.300	—	4.938	—	—	—
Rastrigin(5)	25.30	—	20.52	—	—	—

is proportional to $s_{\max}$ and one can estimate the number of computations that would be executed within $5n + 10$ and $25n + 10$ $s_{\max}$ interval and judge the performance of the method.

With 1.0% accuracy, TRIOPT performs well except for Schwefel and Griewank (4). In particular, its performance in low- and high-dimensional Rastrigin function is notable, where MCS and RBF-O fail and DIRECT requires a much higher number of function evaluation to converge.

When the required accuracy is set to 0.01% (Table 2), it is observed that MCS1 cannot converge with the default tree depth limit. MCS2 converges except in Rastrigin with more than two dimensions. It is observed that MCS requires a high number of function evaluations in Spike, Crescent and Rastrigin(2). At this level of accuracy, the pattern of method performance is similar to that of previous level of accuracy. Namely, methods that perform best in certain functions preserve their ranks. Furthermore, if in a comparison between the number of function evaluations at 1% and 0.01% accuracy levels, a method's requirements do not differ significantly, then, one can deduce that the method had already converged to the global basin of attraction at the 1% accuracy level. In this sense, RBF-O is less affected by the level of accuracy in the problems that it can solve. The number of function evaluations are more or less the same as those of 1% accuracy, except the Crescent that requires a significantly larger number of function evaluations. In Table 2, DIRECT is observed to require a significantly larger number of function evaluations, specifically

in Griewank (3) and Griewank (4). The performance of TRIOPT is less affected by the refined level of accuracy as compared with DIRECT, with the exception of Griewank (4). Furthermore, only DIRECT and TRIOPT are able to converge to the global optimum in all functions at this level of accuracy, though DIRECT needs a significantly larger amount of function evaluations.

Summarizing the results, it is found that TRIOPT is less affected by surface smoothness and continuity around the global optimum, which are assumptions that have to hold for the convergence of other black box optimization methods. Interpolation based methodologies such as RBF-O explicitly make use of this assumption by selecting the new sample location according to the interpolated surface. The box selection criterion utilized by DIRECT assumes a linear acceleration in function value improvements and this is also based on the assumption of smoothness, whereas the maximum gain splitting criterion in MCS is based on a quadratic approximation of the function. On the other hand, TRIOPT utilizes triangulation as an efficient partitioning method without approximating the surface through interpolation. Since the simplex selection criterion and the location of new vertices in the partitioning procedures depend only on the transformed function values of the parent vertices and the resulting entropy, TRIOPT is less affected by nonsmooth surfaces. Furthermore, the method is also efficient in smooth but nondifferentiable surfaces such as the Crescent. The dynamic thresholding scheme of TRIOPT enables less attractive regions to be scanned in parallel with other promising regions in earlier stages of the search and this puts the method at an advantageous position.

In general terms, the efficiency ratio, er, for MCS and DIRECT are 2 (or 3) and 1.5, respectively. For TRIOPT, er equals to $n+1$, and 2, respectively, when splitting is carried out according to interior-simplex and on-edge splitting for *nD* functions. Thus, the number of function evaluations needed for re-partitioning is smaller. Regarding the search scheme, MCS partitions one box at a time, and DIRECT has a parallel splitting strategy based on the criteria pair (function value, box width). Similar to DIRECT, TRIOPT also has a parallel strategy based on entropy. This might sometimes become a disadvantage, because, due to its higher er, TRIOPT produces more partitions that are eligible for re-partitioning, especially in functions where the height of extremities are very close to each other and reflect no trend towards the global optimum. This is the reason why TRIOPT results in significant numbers of function evaluations in Schwefel and Griewank (4). However, the latter might be prevented by limiting the number of parallel search threads as an explicit control mechanism.

4. Conclusion

Black box optimization techniques have an important role in real world applications, especially when a function is expensive to evaluate, or, additional information on the function is difficult to obtain.

A triangulation-based partitioning algorithm, TRIOPT, is proposed here. TRIOPT has a high simplex-based partitioning efficiency and it is integrated within a flexible parallel search scheme based on dynamic thresholding and function transformation. In this scheme, the degree of parallelism (breadth) in the search tree is self-adaptive and based on the status of the search.

Another advantage of TRIOPT over other partitioning algorithms is that if some experimental results (function evaluations) are already obtained from a feasible domain without a specific pattern, an initial partition is easily generated by Delaunay triangulation. Such cases might be more difficult to handle by other partitioning methods that require a careful design of initial samples to minimize the number of function evaluations in further partitioning.

Furthermore, the partition assessment and selection methodology in TRIOPT are not based on a priori assumptions on the function, such as surface smoothness and continuity, and therefore, its performance is more robust when these restrictions are violated.

Acknowledgements

We thank the Editorial Office for the effort made to communicate necessary information throughout the review process.

Appendix A. Brief description of test functions

Spike $\mathbb{R}^2$, [18]: Two-dimensional problem with innumerable local optima, haystack type, very narrow basin of attraction for global optimum, small difference between global and second best optimum.

$$f(\mathbf{x}) = 21.5 + x_1 \sin(4\pi x_1) + x_2 \sin(20\pi x_2), \quad x_1 = [-3.0, 12.1], \quad x_2 = [4.1, 5.8].$$

Ackley $\mathbb{R}^2$, [2]: Symmetric multi-modal test problem with evenly distributed maxima. Original feasible region shifted.

$$f(\mathbf{x}) = 20 \exp(-0.2\sqrt{0.5(x_1^2 + x_2^2)}) + \exp(0.5(\cos(2\pi x_1) + \cos(2\pi x_2))) - 20.0 - e,$$
$$x_1 = [-1.5, 2.5], \ x_2 = [-2.0, 2.0].$$

Schwefel $\mathbb{R}^2$, [28]: Deceptive function, global optimum near domain corner, three second best local optima far away from the global optimum.

$$f(\mathbf{x}) = \sum_{i=1}^{2} x_i \sin(\sqrt{|x_i|}) - 837.9658, \quad x_i = [-500.0, 500.0], \ i = 1, 2.$$

Easom $\mathbb{R}^2$, [9]: A unimodal test problem with a very flat surface outside the basin of attraction.

$$f(\mathbf{x}) = \cos(x_1)\cos(x_2)\exp(-(x_1 - \pi)^2 - (x_2 - \pi)^2), \quad x_i = [0.0, 6.0], \ i = 1, 2.$$

Crescent $\mathbb{R}^2$, [17]: Nondifferentiable, a singular global optimum lies a circular loci leading to a crater-like basin of attraction, smooth elsewhere.

$$f(\mathbf{x}) = -\max\{x_1^2 + (x_2 - 1)^2 + x_2 - 1, -x_1^2 - (x_2 - 1)^2 + x_2 + 1\},$$
$$x_1 = [-1.5, 1.5], \ x_2 = [-0.5, 2.5].$$

Griewank $\mathbb{R}^n$, $n = 2, \ldots, 4$, [31]: Many widespread evenly distributed local optima, increased complexity due to reduced original domain. Original feasible domain shifted.

$$f(\mathbf{x}) = -\frac{1}{4000}\sum_{i=1}^{n} x_i^2 + \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) - 1.0, \quad x_i = [-9.0, 11.0], \ i = 1, \ldots, n.$$

Rastrigin $\mathbb{R}^n$, $n = 2, \ldots, 5$, [31]: Highly multi-modal, evenly spaced haystack type, spherical and symmetric. Original feasible domain shifted.

$$f(\mathbf{x}) = \sum_{i=1}^{n} [10\cos(2\pi x_i) - x_i^2] - 10n, \quad x_i = [-3.12, 5.12], \ i = 1, \ldots, n.$$

References

[1] F. Aurenhammer, Voronoi diagrams—a survey of a fundamental geometric data structure, ACM Comput. Surveys (CSUR) 23 (1991) 345–405.
[2] Thomas Bäck, Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms, Oxford University Press, New York, 1996.
[3] C.B. Barber, D.P. Dobkin, H. Huhdanpaa, The quickhull algorithm for convex hulls, ACM Trans. Math. Software 22 (1996) 469–483.
[4] M. Björkman, K. Holmström, Global optimization of costly nonconvex functions using radial basis functions, Optim. Eng. 1 (2000) 373–397.
[5] C.G.E. Boender, H.E. Romeijn, Stochastic methods, in: R. Horst, P.M. Pardalos (Eds.), Handbook of Global Optimization, Kluwer Academic Publishers, Dordrecht, 1995, pp. 829–869.
[6] P. Cignoni, C. Montani, R. Scopigno, DeWall: a fast divide and conquer Delaunay triangulation algorithm in E^d, Comput. Aided Design 30 (1998) 333–341.
[7] M. Demirhan, L. Ozdamar, A note on the use of a fuzzy approach in adaptive partitioning algorithms for global optimization, IEEE Trans. Fuzzy Systems 7 (1999) 468–475.
[8] C.V. Deutsch, A.G. Journel, GSLIB: Geostatistical Software Library and User's Guide, Oxford University Press, New York, 1998.
[9] Genetic and Evolutionary Algorithm Toolbox for Matlab. http://www.systemtechnik.tu-ilmenau.de/~pohlheim/GA_Toolbox/fcnindex.html.
[10] H.-M. Gutmann, A radial basis function method for global optimization, J. Global Optim. 19 (2001) 201–227.
[11] E. Hansen, Global Optimization Using Interval Analysis, Marcel Dekker, New York, 1992.
[12] P. Hansen, B. Jaumard, Lipschitz optimization, in: R. Horst, P.M. Pardalos (Eds.), Handbook of Global Optimization, Kluwer Academic Publishers, Dordrecht, 1995, pp. 407–493.
[13] W. Huyer, A. Neumaier, Global optimization by multilevel coordinate search, J. Global Optim. 14 (1999) 331–355.
[14] D.R. Jones, Global optimization with response surfaces, Fifth SIAM Conference on Optimization, Canada, 1996.
[15] D.R. Jones, A taxonomy of global optimization methods based on response surfaces, J. Global Optim. 21 (2001) 345–383.
[16] D.R. Jones, C.D. Perttunen, B.E. Stuckman, Lipschitzian optimization without the Lipschitz constant, J. Optim. Theory Appl. 79 (1993) 157–181.
[17] M.M. Mäkelä, P. Neittaanmäki, Nonsmooth Optimization, World Scientific, Singapore, 1992.
[18] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer, Berlin, 1996.
[19] J. Mockus, Application of Bayesian approach to numerical methods of global and stochastic optimization, J. Global Optim. 4 (1994) 347–365.
[20] E. Onbaşoğlu, L. Ozdamar, Parallel simulated annealing algorithms in global optimization, J. Global Optim. 19 (2001) 27–50.
[21] L. Ozdamar, M.B. Demirhan, Experiments with new stochastic global optimization search techniques, Comput. Oper. Res. 27 (2000) 841–865.
[22] L. Ozdamar, M.B. Demirhan, Comparison of partition evaluation measures in an adaptive partitioning algorithm for global optimization, Fuzzy Sets Systems 117 (2001) 47–60.
[23] N.R. Pal, S.K. Pal, Object background segmentation using new definitions of entropy, IEE Proc. 136 (1989) 284–295.
[24] J. Pinter, Convergence qualification of adaptive partition algorithms in global optimization, Math. Programming 56 (1992) 343–360.
[25] J.D. Pinter, Global Optimization in Action, Kluwer Academic Publishers, Dordrecht, 1996.

[26] M.J.D. Powell, The theory of radial basis function approximation in 1990, in: W. Light (Ed.), Advances in Numerical Analysis, vol. II, Wavelets, Subdivision Algorithms and Radial Basis Functions, Oxford University Press, Oxford, 1992, pp. 105–210.
[27] H. Ratschek, J. Rokne, Interval methods, in: R. Horst, P.M. Pardalos (Eds.), Handbook of Global Optimization, Kluwer Academic Publishers, Dordrecht, 1995, pp. 751–828.
[28] H.-P. Schwefel, Numerical Optimization of Computer Models, Wiley, Chichester, 1981.
[29] P. Su, R.L. (Scot) Drysdale III, A Comparison of Sequential Delaunay Triangulation Algorithms, Symposium on Computational Geometry, 1995, pp. 61–70.
[30] A. Törn, M.M. Ali, S. Viitanen, Stochastic global optimization: problem classed and solution techniques, J. Global Optim. 14 (1999) 437–447.
[31] A. Törn, A. Žilinskas, Global Optimization, Lecture Notes in Computer Science, vol. 350, Springer, Berlin, 1989.